|  |  |
| --- | --- |
|  | **FPT ACADEMY INTERNATIONAL**  **FPT – APTECH COMPUTER EDUCATION** |

**Centre Name: ACE-HCMC-2-FPT.**

**Address: 590 Cach Mang Thang 8, District 3, Ho Chi Minh City, Viet Nam.**

**Symphony Limited**

|  |  |  |
| --- | --- | --- |
| **Supervisor:** | MR. NGO PHUOC NGUYEN | |
| **Semester:** | 3 | |
| **Batch No:** | T1.2005.E1 | |
| **Group No:** | 1 | |
| **Order:** | **Full name** | **Roll No.** |
| 1. | Nguyen Thanh Trung | Student1266450 |
| 2. | Ha Tan Phat | Student1248886 |
| 3. | Vu Hoang Thach | Student1248880 |
| 4. | Phan Thi Tuyet Nhung | Student1241103 |

September 2021

This is to certify that

Mr. Nguyen Thanh Trung

Mr. Ha Tan Phat

Mr. Vu Hoang Thach

Ms. Phan Thi Tuyet Nhung



have successfully designed and

developed

Submitted by:

Date of Issue: \_\_\_\_\_\_\_\_

Authorized Signature: \_\_\_\_\_\_\_\_\_\_\_

**Table of Contents**

[ACKNOWLEDGMENT 7](#_Toc83743611)

[1. Introduction 8](#_Toc83743612)

[2. Existing Scenario 8](#_Toc83743613)

[3. Customer Requirement Specification 10](#_Toc83743614)

[4. Functional Requirements 10](#_Toc83743615)

[4.1. Admin Can Manage Courses 10](#_Toc83743616)

[4.2. Admin Can Manage Examinations 12](#_Toc83743617)

[4.3. Admin Can Manage Student’s Information 12](#_Toc83743618)

[4.4. Admin Can Manage User’s Site Information 12](#_Toc83743619)

[4.5. User Can Enroll on Courses 13](#_Toc83743620)

[4.6. User Can Take Examinations 13](#_Toc83743621)

[4.7. User Can Views Study Results and Lab Sessions 14](#_Toc83743622)

[HARDWARE/ SOFTWARE REQUIREMENTS 15](#_Toc83743623)

[1. Server Requirement 15](#_Toc83743624)

[2. Client Requirement 16](#_Toc83743625)

[3. Software and technologies used for project 16](#_Toc83743626)

[TASK SHEET REVIEW 1 17](#_Toc83743627)

[Use Cases and Sequence Diagrams 19](#_Toc83743628)

[1. User 19](#_Toc83743629)

[1.1. Register as Student 19](#_Toc83743630)

[1.2. Login as Student: 21](#_Toc83743631)

[1.1. Update Student’s Profile 22](#_Toc83743632)

[1.3. Change Student’s Password: 24](#_Toc83743633)

[1.4. Forget Password: 25](#_Toc83743634)

[2. Course 27](#_Toc83743635)

[2.1 View Course: 27](#_Toc83743636)

[2.2 Search Course: 28](#_Toc83743637)

[2.3 Register for a course: 29](#_Toc83743638)

[2.4 Register for Lab Sessions: 30](#_Toc83743639)

[3. Examination: 31](#_Toc83743640)

[3.1. User view examination details 32](#_Toc83743641)

[3.2. User can register for an examination in advanced course 32](#_Toc83743642)

[3.3. User can take exam 33](#_Toc83743643)

[3.4. User view learning result 34](#_Toc83743644)

[4. Admin 35](#_Toc83743645)

[4.1. User Management 35](#_Toc83743646)

[4.2. Course Management 37](#_Toc83743647)

[4.3. Examination Management 42](#_Toc83743648)

[4.4. Registration Management 47](#_Toc83743649)

[4.5. Other Admin Functionalities 49](#_Toc83743650)

[Data Flow Diagram 50](#_Toc83743651)

[1. Context Diagram: 50](#_Toc83743652)

[2. Level 0 DFD 50](#_Toc83743653)

[3. Level 1 DFD 51](#_Toc83743654)

[Entity–Relationship Diagram 52](#_Toc83743655)

[1. Conceptual ERD 52](#_Toc83743656)

[2. Logical ERD 53](#_Toc83743657)

[3. Physical ERD 54](#_Toc83743658)

[Entities and Properties 55](#_Toc83743659)

[1. AppUsers 55](#_Toc83743660)

[2. Roles 56](#_Toc83743661)

[3. AppRoles 56](#_Toc83743662)

[4. RoleClaims 56](#_Toc83743663)

[5. UserClaims 57](#_Toc83743664)

[6. UserLogins 57](#_Toc83743665)

[7. UserTokens 58](#_Toc83743666)

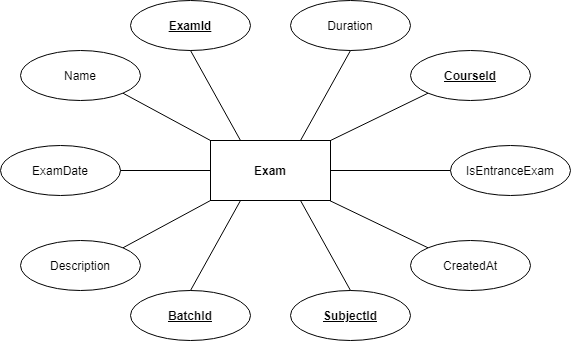
[8. Batch 58](#_Toc83743667)

[9. Course 59](#_Toc83743668)

[10. Subject 59](#_Toc83743669)

[11. Subject\_Course 60](#_Toc83743670)

[1. Exam 61](#_Toc83743671)

[ 61](#_Toc83743672)

[1. CourseRegistration 61](#_Toc83743673)

[1. ExamRegistration 62](#_Toc83743674)

[2. ConsultRegistration 62](#_Toc83743675)

[3. Enrollment 62](#_Toc83743676)

[4. PaymentStatus 63](#_Toc83743677)

**REVIEW I**

# ACKNOWLEDGMENT

First, we would like to thank all those who have given support and help us make this project a success.

Second, we wish to express our gratitude deeply to our teachers who provided us with invaluable advice and helped us in difficult periods.

Besides, we thank our families, our friends from the bottom of our heart for their encouragement and their care. Without their help we couldn’t have succeeded in finishing this project.

We are also thankful to the entire staff and chairpersons at the Head Office of Aptech Worldwide, Aptech – Vietnam, and FPT – Aptech Centre who have been organizing and supporting our course.

There are no words to show our appreciation for the teachers of FPT Aptech Centre who have been organizing and helping us during our course to finish this project perfectly, especially, our teacher Mr Ngo Phuoc Nguyen who teaches us, guides us and gives us useful advice.

Finally, we are really grateful to our school fellows for their valuable suggestions.

Thanks again to all who helped us!

**PROBLEM DEFINITION**

## Introduction

Symphony Ltd. is private institute, which generally conducts the classes and training

materials for the various certifications that are related to IT and Software industries like

networking related, Java related, database related like for SQL Servers, oracle, etc. They

are one of the famous institutions which have spread into various branches situated at

different location in the state.

They provide various resources like the preparation materials, lab facilities based on the

subject, extra lab hours or sessions (offered after the course duration), 24 hours faculty

guidance is provided for the lab sessions, assignments, Library facilities, etc.

## Existing Scenario

Currently they advertise about the services that they provide through the pamphlets (available at the centres), advertisements in newspapers, and through other media like through television advertisements, etc. in order to attract the people. Usually, they conduct the training on a half yearly basis every year.

They generally conduct an entrance exam through which they can know the knowledge level of the participant and thus depending on the exam marks they segregate the participants or the students into the appropriate classrooms like for the participant not having the minimum knowledge will be given training right from the basics till the certification level and for the participants having the minimum knowledge about the basics will be segregated in to the class where they direct start with the basic certification topics without touching the basic concepts of the subject.

Once the examination results are out, they inform the participants with the results, and ask them to visit the institution on a particular date for counselling purposes, where they inform the students with which class they will be segregated and regarding the course fees based on the class segregated. If the student is interested in taking the class, then he will be provided with the class form in which he/she is segregated and then they are directed to the accounts department where the student needs to make the payment in order to join the class. The student can make the payment for the class, at least 1 day before the commencement of the class.

For the training with the basics, they usually charge around 6000$, and the duration for the training will be for 6 months. And for the training without the basics, they charge around 4275$, and the duration will be for 4 months.

They also provide the practice sessions (extra lab sessions) separate from the course duration, i.e., after the successful completion of the course if the student wishes to practice the lab sessions at the lab in the institution, he (student) can prefer at the time of registering for the course. And for this they charge around 1000$ separately from the course fees. But the student has to opt for it at the time of getting registered for the course itself. Else if a student wants to opt for it at the end of the course based on his requirement, then they will charge 1500$.

Note: The extra lab sessions are only available for the students who have registered for the course offered by the institution. And for this they provide guidance too by one of the faculty members.

After the completion of the certification course, they conduct the final examination and based on the marks secured the respective faculty members provides the suggestions and feedback like what are areas of improvement, where extra practice is to be delivered, which other materials apart from the course materials can be preferred, etc. Thus, ensuring that they pay maximum attention to the students and making the student benefit from joining this institution.

For going into the library that the institution provides, the student must display his identity card. And they do not allow the student to carry the library books to his / her house.

## Customer Requirement Specification

* The web application allows the users to browse and search for courses using teams or filtering course categories and attributes.
* If users want to enrol a course, they must sign in to do so. A user with an existing account can login. They do not have the permission to change any information apart from their own.
* If users enrol for a basic course then they will receive basic information regarding the course registration like registration code. Then they can bring that code to the Institute to make a payment for that course.
* If users enrol for an advanced course then they will have to pass an entrance examination first.
* The web application allows registered users to view their study results and upcoming examination.
* The web application must allow admins to manage courses, examinations and some information about the web app like in About Us and F.A.Q page.
* Admin also allows to manage user’s registration and payment status along with it.

## Functional Requirements

### Admin Can Manage Courses

* **Description**

Admin can view the list of courses and click on each to view its details, insert new courses, update existing courses and remove courses.

* **Functional Requirements**

To insert or update a course, admin must provide:

* Course name.
* Description.
* Start and end date.
* Basic or advanced course.

### Admin Can Manage Examinations

* **Description**

Admin can view the list of examinations and click on each to view its details, insert new examinations, update existing examinations and remove examinations.

* **Functional Requirements**

To insert or update an exam, admin must provide:

* Exam name.
* Description.
* Start date.
* Id of batch.
* Id of subject.
* Entrance or normal exam.

### Admin Can Manage Student’s Information

* **Description**

Admin can view the list of students and click on each to view their details, insert new students, update existing students and remove students.

* **Functional Requirements**

To insert or update a student, admin must provide:

* Student’s first and last name.
* Email Address.
* Day of birth.
* Gender.

### Admin Can Manage User’s Site Information

* **Description**

Admin can view current information on the user’s site pages like About Us and FAQ, insert new information, update existing one and remove information.

* **Functional Requirements**

To insert or update information on about us page, admin must provide:

* Content.
* Show status.

To insert or update information on FAQ page, admin must provide:

* Question.
* Answer.
* Show status.

### User Can Enroll on Courses

* **Description**

Users can view all the courses in the course page and click on each course to view its details. Then the user can click on enrol to apply for that course.

If it is a basic course then users will be provided with more information for the registration.

If it is an advanced course then users will have to pay and take part in an entrance examination. After passing that exam, users will be provided with more information for the registration.

* **Functional Requirements**

To enrol on a course, user have to:

* Register an account and fill in all the personal information.
* Choose a course for enrol.
* Click on the enroll button.
* Pay and take part in entrance examinations if required.

### User Can Take Examinations

* **Description**

After users register an account and sign in, he or she can view current entrance examinations and upcoming ones and pay a fee to participate in one.

Students can view their upcoming examinations and take part in it.

* **Functional Requirements**

To take examinations, user have to:

* Register an account.
* Pay for an entrance examination or enroll in a course.

### User Can Views Study Results and Lab Sessions

* **Description**

After a user enrolled on a course, he or she can access the student page.

Students can check their result, existing examinations by entering the student page.

Students also can use their lab session during their course running time in the student page and the lab session is only available before 9pm each day.

Only students who registered for a premium lab session can access it after their course finished.

* **Functional Requirements**

To view study results and lab sessions:

* Users must register an account.
* Finished the enrollment process.

# HARDWARE/ SOFTWARE REQUIREMENTS

## Server Requirement

* + **Hardware**

|  |  |
| --- | --- |
| **Component** | **Requirement** |
| **CPU** | Processor type:  Pentium IV-compatible processor or faster  Processor speed:  Recommended: 2.0 GHz or faster |
| **OS** | Microsoft Windows Server 2012 with IIS |
| **Memory (RAM)** | RAM:  Minimum: 512 MB  Recommended: 2 GB or more |
| **Hard Drive** | Free space:  Minimum: 200 MB  Recommended: 50 GB or more  Maximum: Operating system maximum |

* + **Software**

|  |  |
| --- | --- |
| **Component** | **Requirement** |
| **Microsoft .NET Framework** | Version 4.6 or higher |
| **RDBMS** | Microsoft SQL Server 2017 |

## Client Requirement

* + **Hardware**

|  |  |
| --- | --- |
| **Component** | **Requirement** |
| **CPU** | Processor type:  Pentium III-compatible processor or faster  Processor speed:  Recommended: 1.0 GHz or faster |
| **OS** | All OS (Windows, Linux, Android, Mac OS …) |
| **Memory (RAM)** | RAM:  Minimum: 512 MB  Recommended: 1 GB or more  Maximum: Operating system maximum |
| **Hard Drive** | Free space:  Minimum: 10 MB |

* + **Software**

|  |  |
| --- | --- |
| **Component** | **Requirement** |
| **Web Browser** | Chrome, Opera (up to date) |

## Software and technologies used for project

* **Software:**
* Visual Studio 2019.
* Visual Studio Code.
* Microsoft Sql Server 2019.
* Asp Net Core.
* **Technologies:**
* Asp net core.
* Blazor Server Side.
* Html, Css.

# TASK SHEET REVIEW 1

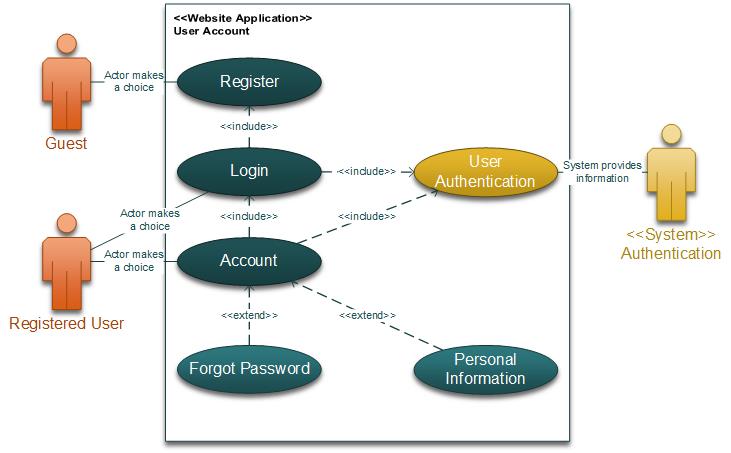
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Project Ref. No: 4 | Project Title:  Railway Reservation Manage System | Date of Preparation of Activity Plan | | | |
| No. | Task | Actual Start Date | Actual Days | Team Member Names | Status |
| 01 | Acknowledgements | 11, 09 2021 | 2 | Nguyen Thanh Trung | Completed |
| 02 | Introduction | Nguyen Thanh Trung | Completed |
| 03 | Existing Scenario | Nguyen Thanh Trung | Completed |
| 04 | Customer Requirement | Ha Tan Phat | Completed |
| 05 | Functional Requirement | Vu Hoang Thach | Completed |
| 06 | Hardware & Software Requirements | Phan Thi Tuyet Nhung | Completed |
| 07 | Task sheet | Phan Thi Tuyet Nhung | Completed |

|  |  |  |
| --- | --- | --- |
|  | Prepare By: Group 1 | Approved By: Faculty |
| Date: 21, 09, 2021 | Team Leader  Nguyen Thanh Trung | Mr. Ngo Phuoc Nguyen |

**REVIEW II**

# Use Cases and Sequence Diagrams

## User



### Register as Student

|  |  |  |
| --- | --- | --- |
| **Author** | Ha Tan Phat | |
| **Use Case Name** | Register as Student | |
| **Actors** | Guest | |
| **Description** | Guest registers an account on the website and becomes a member (student) | |
| **Requirements** | Guest provides their register information | |
| **Pre-conditions** | N/A | |
| **Post-conditions** | Success: Create account successfully | |
| Fail: Refill information | |
| **Basic flow** | **Actor’s actions:** | **System’s responses:** |
| 1. Actor clicks ‘Register’ on the Home Page.  3. Actor inputs register’s information and click ‘Register’ button.  7. User goes to email account and click on confirm. | 2. System redirects to Register page and displays Registration form with the following controls:   * **‘**First Name’ text field * ‘Last Name’ text field * ‘Phone Number’ text field * ‘Gender’ radio button * ‘Day of Birth’ date selector * ‘Email’ text field * ‘Password’ text field * ‘Confirm Password’ text field * ‘Submit and Reset’ button   4. System validates the information.  5. System inserts the account into database.  6. System ask the user to confirm their email account.  8. System display confirm email successfully. |
|
|
| **Exceptions** | **Actor’s actions:** | **System’s responses:** |
| - Actor left ‘First Name’ text field blank.  - Actor left ‘Last Name’ text field blank.  - Actor left ‘Phone Number’ text field blank.  - Actor left ‘Email’ text field blank.  - Actor left ‘Password’ text field blank.  - Actor left ‘Confirm Password’ text field blank.  - Actor enter an existed Email address.  - Actor enter an incorrect confirm password. - Input age is less than 16.  - Input phone number is not numeric format or phone length is not between 8 and 14 digits. | System displays message to actor:   * ‘The First Name field is required’. * ‘The Last Name field is required’. * ‘The Phone Number field is required’. * ‘The Email field is required’. * ‘The Password field is required’. * ‘The Confirm Password field is required’. * ‘Email is already taken’. * ‘Confirm Password does not match password.’ * ‘Age must be over 16’. * ‘The phone number should be numeric only and must be between 8 and 14 digits. |

## 

### Login as Student:

|  |  |  |
| --- | --- | --- |
| **Author** | Ha Tan Phat | |
| **Use Case Name** | Login | |
| **Actors** | Student | |
| **Description** | Student who has registered an account can login | |
| **Requirements** | Actor provides email and password | |
| **Pre-conditions** | Actor must register an account | |
| **Post-conditions** | Success: User is logged in to website | |
| Fail: Refill information | |
| **Basic flow** | **Actor’s actions:** | **System’s responses:** |
| 1. Actor clicks on ‘Login’ button.     3. Actor inputs Email and Password, then click the ‘Login’ button. | 2. System redirects to Login Page with the following controls - ‘Email’ text field - 'Password' text field - 'Login' button 4. System checks the information. 5. System redirects to User’s Dashboard. |
|
|
| **Exceptions** | **Actor’s actions:** | **System’s responses:** |
| - Actor left ‘Email’ text field blank.  -Actor left ‘Password’ text field blank.  - Actor inputs invalid email.  - Actor inputs invalid password. | System shows message:  - ‘The Email field is required’.  - ‘The Password field is required’.  - ‘Invalid Email’.  - ‘Invalid Password’. |

### Update Student’s Profile

|  |  |  |
| --- | --- | --- |
| **Author** | Ha Tan Phat | |
| **Use Case Name** | Update Student’s Profile | |
| **Actors** | Student | |
| **Description** | Student wants to Update his/her profile’s information. | |
| **Requirements** | Actor provides profile information | |
| **Pre-conditions** | Actor has logged in. | |
| **Post-conditions** | Success: Profile’s information of this user has been updated. | |
| Fail: Refill information | |
| **Basic flow** | **Actor’s actions:** | **System’s responses:** |
| 1. Actor clicks ‘Profile’ button in Actor’s Home page menu.         3. Actor updates information and click ‘Save’ button. | 2. System redirects to Update Profile page and displays Update Profile form with the following controls:   * ‘First Name’ text field * ‘Last Name’ text field * ‘Phone Number’ text field * ‘Gender’ radio buttons * ‘Day of Birth’ date selector. * ‘Address’ text field. * ‘Save’ button. * ‘Cancel’ button.   4. System checks the information.  5. System updates the account in database and then redirects to User’s Dashboard |
|
| **Exceptions** | **Actor’s actions:** | **System’s responses:** |
| - Actor left ‘First Name’ text field blank.  - Actor left ‘Last Name’ text field blank.  - Actor left ‘Phone Number’ text field blank.  - Actor left ‘Address’ text field blank.  - Input age is less than 16.  - Input phone number is not numeric format or phone length is not between 8 and 14 digits. | System displays message to actor:   * ‘The First Name field is required’. * ‘The Last Name field is required’. * ‘The Phone Number field is required’. * ‘The Address field is required’. * ‘Age must be over 16’. * ‘The phone number should be numeric only and must be between 8 and 14 digits. |

### 

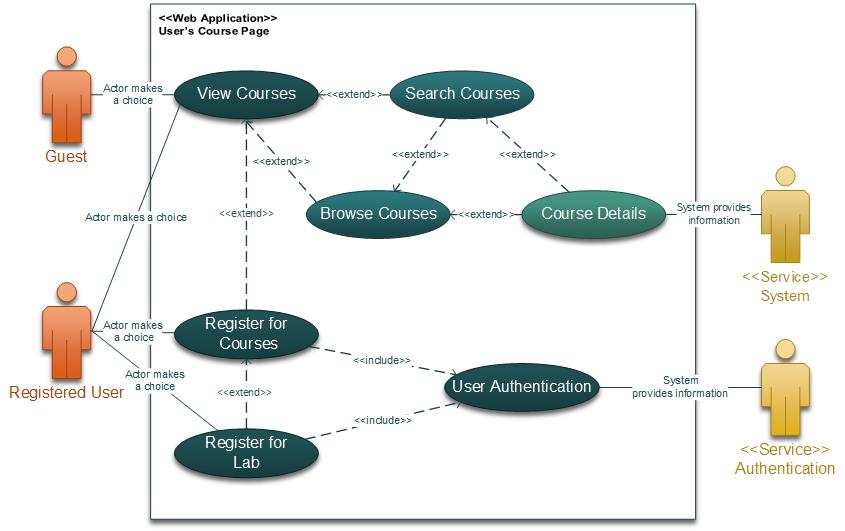
### Change Student’s Password:

|  |  |  |
| --- | --- | --- |
| **Author** | Ha Tan Phat | |
| **Use Case Name** | Change Password | |
| **Actors** | Student | |
| **Description** | Student who wants to change password | |
| **Requirements** | Actor provides current password and new password | |
| **Pre-conditions** | Actor has logged in. | |
| **Post-conditions** | Success: Actor can change their password | |
| Fail: Refill information | |
| **Basic flow** | **Actor’s actions:** | **System’s responses:** |
| 1. Actor clicks ‘Edit Profile’ tag at ‘Profile’ button on Homepage. And then click “Change Password” button.      3. Actor inputs all text field and clicks ‘Update’ button. | 2. System redirects to Password Page with the following controls:  - 'Current Password' text field  - 'New Password' text field  - 'Confirm Password' text field  4. System checks the information.  5. System updates new password of account in database and then redirects to User Profile Page. |
|
| **Exceptions** | **Actor’s actions:** | **System’s responses:** |
| - Actor left ‘Current Password’ field blank.  - Actor inputs wrong ‘Current Password’.  - Actor left ‘New Password’ field blank.  - Actor left ‘Confirm Password’ text field blank.  - New password and confirm password don’t match. | System shows messages:  - ‘ The Current Password field is required’.  - ‘Current password was wrong’.  - ‘The New Password field is required’.  - ‘Confirm Password cannot be blank’.  - ‘Confirm password does not match.’ |

### Forget Password:

|  |  |  |
| --- | --- | --- |
| **Author** | Ha Tan Phat | |
| **Use Case Name** | Forget Password | |
| **Actors** | Student | |
| **Description** | Student who has forgotten his/her password can make a request to change his/her password | |
| **Requirements** | Actor provides email address | |
| **Pre-conditions** | Actor must have an account | |
| **Post-conditions** | Success: Actor can change their password | |
| Fail: Refill information | |
| **Basic flow** | **Actor’s actions:** | **System’s responses:** |
| 1. Actor clicks on ‘Forgot Password’ button.      3. Actor inputs Email and clicks ‘Submit’.    5. Actor inputs reset token in Password Page and click ‘Submit’.        7. Actor inputs ‘New Password’, ‘Confirm Password’ and then click ‘Save’. | 2.System redirects to Password Page with the following controls:   * **'Email' text field** * **'Submit' button**   4. System checks the information and sends a reset token to user.  6. System redirects to Password form with the following controls:  - ‘New Password’ text field  - ‘Confirm Password’ text field  - ‘Save’ button  8. System validates password then updates password in database.  9. System shows message: ‘Your password has been changed successfully’. |
|
| **Exceptions** | **Actor’s actions:** | **System’s responses:** |
| - Actor left ‘Email’ text field blank.  - Actor inputs an email which does not exist in database.  - Actor inputs authentication code incorrectly.  -Actor left ‘New Password’ field blank.  - Actor left ‘Confirm Password’ field blank.  -Actor inputs ‘Confirm Password’ that does not match ‘New Password’. | System shows messages:  - The Email field is required’.  - ‘Invalid email’.  - ‘Invalid authentication code’.  - ‘The New Password field is required’.  - ‘The Confirm Password field is required’.  - ‘Confirm Password does not match New Password.’ |

## Course



### View Course:

|  |  |  |
| --- | --- | --- |
| **Author** | **Nguyen Thanh Trung** | |
| **Use Case Name** | View courses by subjects | |
| **Actors** | Guest/Student | |
| **Description** | Guest/Student wants to browse courses | |
| **Requirements** | Actor provides email address | |
| **Pre-conditions** | N/A | |
| **Post-conditions** | Success: Show all courses or filtered courses | |
| Fail: N/A | |
| **Basic flow** | **Actor’s actions:** | **System’s responses:** |
| 1. Actor clicks on Courses on the navigation bar.  3. Actor types on bar: course name or subject name or duration, then clicks filter.  5. Actor clicks on a course | 2. System shows a page that includes:  - A bar text field to filter.  - A “Filter” button  - List all courses  4. System displays all courses that match this filter:  6. System shows more details about this course. |
|

### Search Course:

|  |  |  |
| --- | --- | --- |
| **Author** | **Nguyen Thanh Trung** | |
| **Use Case Name** | Search courses by name | |
| **Actors** | Guest/Student | |
| **Description** | Guest/Student wants to search for courses by name | |
| **Requirements** | Actor provides search phrase | |
| **Pre-conditions** | N/A | |
| **Post-conditions** | Success: Show courses that matches search phrase | |
| Fail: No course is shown | |
| **Basic flow** | **Actor’s actions:** | **System’s responses:** |
| 1. Actor types in search phrase and clicks ‘Search’ button.  3. Actor clicks on a course name in this list. | 2. System shows the list of courses that match this search phrase.  4. System display more details about this course. |
|

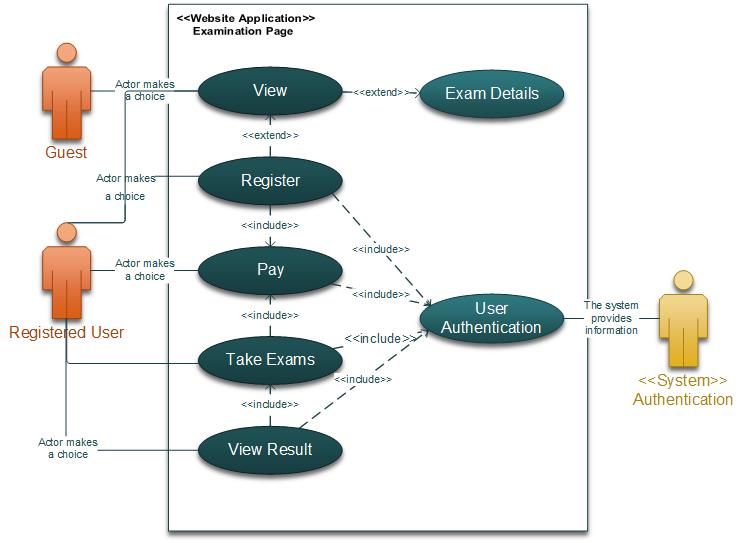
### Register for a course:

|  |  |  |
| --- | --- | --- |
| **Author** | **Nguyen Thanh Trung** | |
| **Use Case Name** | Register for a course | |
| **Actors** | Student | |
| **Description** | Student wants to register for a course | |
| **Requirements** | Student must have an account | |
| **Pre-conditions** | Actor has logged in | |
| **Post-conditions** | Success: Actor join a course or join an entrance examination | |
| Fail: Take the entrance examination again or join basic course | |
| **Basic flow** | **Actor’s actions:** | **System’s responses:** |
| 1. Actor fills in address if he/she does not have one.    3. Actor selects a payment method     5. Actor fills in the note in the form (optional)   7. Actor inputs order information, then click ‘Place Order’ button | 2. System saves his/her address to Order in the database   4. System saves payment method to Transaction in the database   6. System saves note to Order in the database   8. System redirects to Order page and shows order’s status. |
|
| **Alternative flow** | Actor goes back to cart or product page | System redirects to the cart or product page and keep the product in the cart. |
| **Exceptions** | **Actor’s actions:** | **System’s responses:** |
| - Actor left ‘Address’ field blank.  - Actor closes the website without clicking ‘Place Order’ button. | - System responses a message ‘The Address field is required’.  - System keeps this cart information in cookies and database. |

### Register for Lab Sessions:

|  |  |  |
| --- | --- | --- |
| **Author** | **Nguyen Thanh Trung** | |
| **Use Case Name** | Register for Lab Sessions | |
| **Actors** | Student | |
| **Description** | Student wants to use lab session | |
| **Requirements** | Register for extra lab session to use it after 9pm | |
| **Pre-conditions** | Actor has logged in | |
| **Post-conditions** | Success: Actor can access the lab session | |
| Fail: Access denied | |
| **Basic flow** | **Actor’s actions:** | **System’s responses:** |
| 1. Actor opens the student corner page.  3. Actor select     5.  7. | 2. System shows the pages with the name of the lab session.  4. System saves payment method to Transaction in the database   6.  8. System redirects to Order page and shows order’s status. |
|
| **Exceptions** | **Actor’s actions:** | **System’s responses:** |
| - Actor left ‘Address’ field blank.  - Actor closes the website without clicking ‘Place Order’ button. | - System responses a message ‘The Address field is required’.  - System keeps this cart information in cookies and database. |

## Examination:



### User view examination details

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use Case Name** | View Examination details | |
| **Actors** | Student | |
| **Description** | Student wants to view examination details. | |
| **Requirements** | Student must have an account. | |
| **Pre-conditions** | Actor has logged in. | |
| **Post-conditions** | Success: Actor can see examinations. | |
| N/A | |
| **Basic flow** | **Actor’s actions:** | **System’s responses:** |
| 1. Actor clicks ‘Examinations’. 3. Actor clicks a examination. | 2. System shows a page that has:  - A list of exams.  4. System displays the information of this examination. |
|

### 

### User can register for an examination in advanced course

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use Case Name** | Register for an examination | |
| **Actors** | Student | |
| **Description** | Student wants to Register for an examination in advanced course. | |
| **Requirements** | Student must have an account. | |
| **Pre-conditions** | Actor has logged in. | |
| **Post-conditions** | Success: Register successfully. | |
| Fail: N/A | |
| **Basic flow** | **Actor’s actions:** | **System’s responses:** |
| 1. Actor clicks ‘Register’ button in Examination Detail Page for an advanced course. | 2. System saves information in database and sends the information about this examination through student’s email: student name, student ID, password to take exam, date for exam, time for exam. |
|

### 

### User can take exam

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use Case Name** | Take exam | |
| **Actors** | Student | |
| **Description** | Student wants to take exam. | |
| **Requirements** | Student must have an account.  Student must be in a course or register for an examination of an advanced course. | |
| **Pre-conditions** | Actor has logged in. | |
| **Post-conditions** | Success: | |
| Fail: | |
| **Basic flow** | **Actor’s actions:** | **System’s responses:** |
| 1. Actor clicks ‘Take exam’.     3. Actor inputs ‘Password for exam’ and then clicks ‘Go’ button    5.Actor does this exam by choosing a correct answer. When actor have done this exam, clicks ‘Finish’ button | 2. System redirects to Take Exam page and displays form with the following controls:   * ‘Password for exam’ text field. * ‘Go’ button.   4. System shows the exam.  6. System checks all the answers, then shows the result of this exam. |
|
| **Exceptions** | **Actor’s actions:** | **System’s responses:** |
| - Actor left ‘Password for exam’.  - Input wrong ‘Password for exam’.  - Actor does not click ‘Finish’ button when time out. | System displays message to actor:   * ‘The Password for exam field is required’. * ‘Wrong password for exam’. * System automatically stops what actor is doing, checks answers and shows result. |

### 

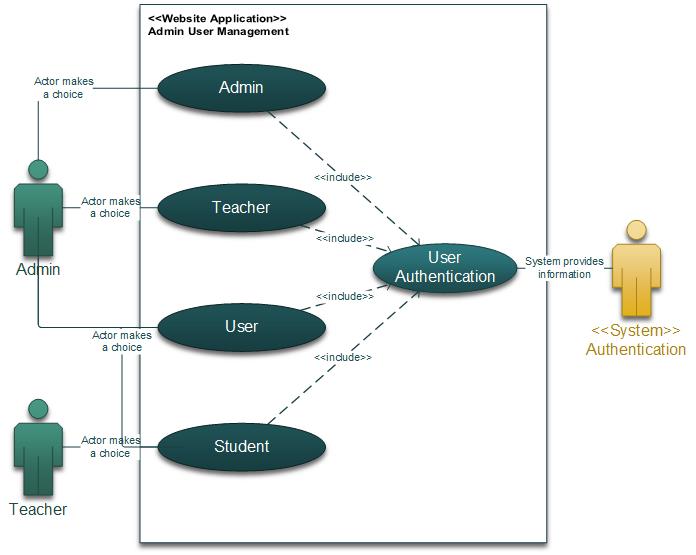
### User view learning result

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use Case Name** | View learning result | |
| **Actors** | Student | |
| **Description** | Student wants to view learning result. | |
| **Requirements** | Student must have an account. | |
| **Pre-conditions** | Actor has logged in. | |
| **Post-conditions** | Success: Actor sees a table of learning result. | |
| N/A | |
| **Basic flow** | **Actor’s actions:** | **System’s responses:** |
| 1. Actor clicks ‘Result’. | 2. System shows a table of learning result. |
|

### 

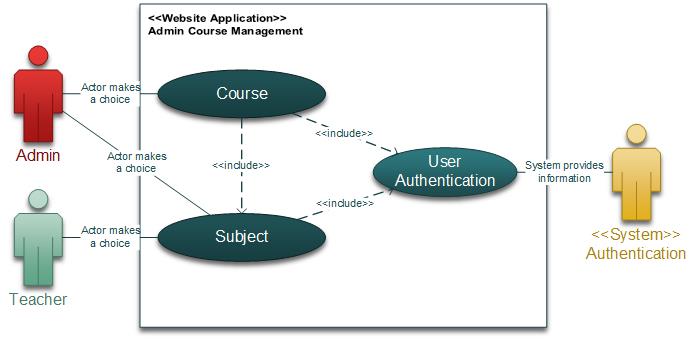
## Admin

### User Management



|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use Case Name** |  | |
| **Actors** | Admin | |
| **Description** |  | |
| **Requirements** |  | |
| **Pre-conditions** |  | |
| **Post-conditions** | Success: | |
| Fail: | |
| **Basic flow** | **Actor’s actions:** | **System’s responses:** |
|  |  |
|
| **Exceptions** | **Actor’s actions:** | **System’s responses:** |
|  |  |

### Course Management



* + 1. View courses

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use Case Name** | View courses | |
| **Actors** | Admin | |
| **Description** | Admin can view courses | |
| **Requirements** | N/A | |
| **Pre-conditions** | Actor has logged in. | |
| **Post-conditions** | Success: View all courses | |
| Fail: N/A | |
| **Basic flow** | **Actor’s actions:** | **System’s responses:** |
| 1. Actor clicks Courses on Dashboard Page. | 2. System shows all courses |
|

* + 1. **View course details**

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use Case Name** | View course details | |
| **Actors** | Admin | |
| **Description** | Admin can view course details | |
| **Requirements** | N/A | |
| **Pre-conditions** | Actor has logged in. | |
| **Post-conditions** | Success: View details of a course | |
| Fail: N/A | |
| **Basic flow** | **Actor’s actions:** | **System’s responses:** |
| 1. Actor clicks any course on Courses Page. | 2. System shows the information of this course. |
|

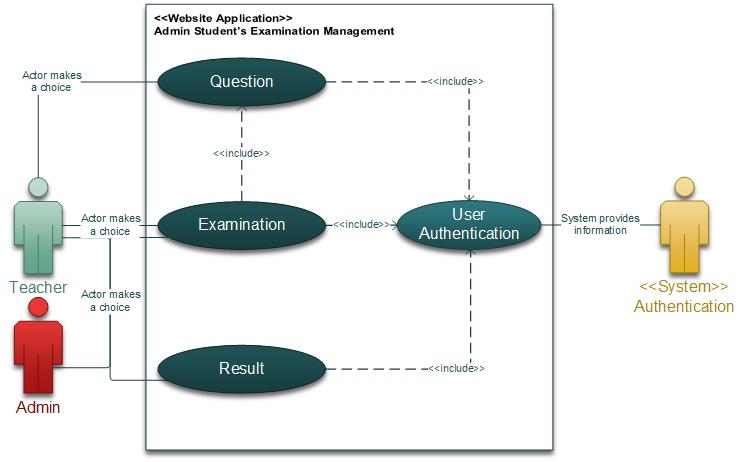
* + 1. Create new course

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use Case Name** | Create new course | |
| **Actors** | Admin | |
| **Description** | Admin can create new course | |
| **Requirements** | N/A | |
| **Pre-conditions** | Actor has logged in. | |
| **Post-conditions** | Success: Course is created successfully | |
| Fail: Refill information | |
| **Basic flow** | **Actor’s actions:** | **System’s responses:** |
| 1. Actor clicks ‘New Course’ button on Courses Page.        3. Actor inputs Course’s information and clicks ‘Create’ button. | 2. System shows a create course form with the following controls:  - ‘Course Name’ text field.  - ‘Description’ text field.  - ‘Start Date’ date selector.  - ‘End Date’ date selector.  - ‘Discount Price’  - ‘Is Extra’  - ‘Is Basic’  - ‘Price’ text field.  - ‘Is Shown’  4. System checks the information.  5. System redirects to Courses Page. |
|
| **Exceptions** | **Actor’s actions:** | **System’s responses:** |
| - Actor does not input Course name.  - Actor does not input Description.  - Actor does not input Price.  - Actor inputs the existing course name. | System displays message to actor:   * ‘The Course Name field is required’. * ‘The Description field is required’. * ‘The Price field is required’. * ‘The course name has existed’. |

* + 1. Edit course information

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use Case Name** | Edit course information | |
| **Actors** | Admin | |
| **Description** | Admin can edit course information | |
| **Requirements** | Admin must have courses to edit | |
| **Pre-conditions** | Actor has logged in. | |
| **Post-conditions** | Success: Course is edited successfully | |
| Fail: Refill information | |
| **Basic flow** | **Actor’s actions:** | **System’s responses:** |
| 1. Actor clicks ‘Edit Course’ button on ‘Course Detail’ Page.        3. Actor inputs updated information and clicks ‘Save’ button. | 2. System shows a course detail with the following controls:  - ‘Course Name’ text field.  - ‘Description’ text field.  - ‘Start Date’ date selector.  - ‘End Date’ date selector.  - ‘Discount Price’  - ‘Is Extra’  - ‘Is Basic’  - ‘Price’ text field.  - ‘Is Shown’  4. System checks the information.  5. System redirects to Courses Page. |
|
| **Exceptions** | **Actor’s actions:** | **System’s responses:** |
| - Actor does not input Course name.  - Actor does not input Description.  - Actor does not input Price.  - Actor inputs the existing course name. | System displays message to actor:   * ‘The Course Name field is required’. * ‘The Description field is required’. * ‘The Price field is required’. * ‘The course name has existed.’ |

### Examination Management



* + 1. View examination

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use Case Name** | View examinations | |
| **Actors** | Admin | |
| **Description** | Admin can view examinations | |
| **Requirements** | N/A | |
| **Pre-conditions** | Actor has logged in. | |
| **Post-conditions** | Success: View all examinations | |
| Fail: N/A | |
| **Basic flow** | **Actor’s actions:** | **System’s responses:** |
| 1. Actor clicks Examination on Dashboard Page. | 2. System shows all examinations. |
|

* + 1. View examination details

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use Case Name** | View examination details | |
| **Actors** | Admin | |
| **Description** | Admin can view examination details | |
| **Requirements** | N/A | |
| **Pre-conditions** | Actor has logged in. | |
| **Post-conditions** | Success: View details of a examination | |
| Fail: N/A | |
| **Basic flow** | **Actor’s actions:** | **System’s responses:** |
| 1. Actor clicks on any examination on Examination Page. | 2. System shows the information of this examination. |
|

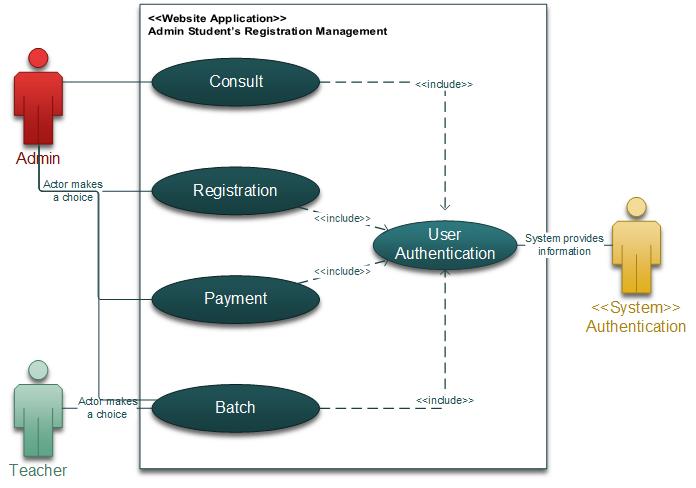
* + 1. Create examination

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use Case Name** | Create new examination | |
| **Actors** | Admin | |
| **Description** | Admin can create new examination | |
| **Requirements** | N/A | |
| **Pre-conditions** | Actor has logged in. | |
| **Post-conditions** | Success: Course is created successfully | |
| Fail: Refill information | |
| **Basic flow** | **Actor’s actions:** | **System’s responses:** |
| 1. Actor clicks ‘New Course’ button on Courses Page.        3. Actor inputs Course’s information and clicks ‘Create’ button. | 2. System shows a create course form with the following controls:  - ‘Course Name’ text field.  - ‘Description’ text field.  - ‘Start Date’ date selector.  - ‘End Date’ date selector.  - ‘Discount Price’  - ‘Is Extra’  - ‘Is Basic’  - ‘Price’ text field.  - ‘Is Shown’  4. System checks the information.  5. System redirects to Courses Page. |
|
| **Exceptions** | **Actor’s actions:** | **System’s responses:** |
| - Actor does not input Course name.  - Actor does not input Description.  - Actor does not input Price.  - Actor inputs the existing course name. | System displays message to actor:   * ‘The Course Name field is required’. * ‘The Description field is required’. * ‘The Price field is required’. * ‘The course name has existed’. |

* + 1. Edit examination

|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use Case Name** | Edit course information | |
| **Actors** | Admin | |
| **Description** | Admin can edit course information | |
| **Requirements** | Admin must have courses to edit | |
| **Pre-conditions** | Actor has logged in. | |
| **Post-conditions** | Success: Course is edited successfully | |
| Fail: Refill information | |
| **Basic flow** | **Actor’s actions:** | **System’s responses:** |
| 1. Actor clicks ‘Edit Course’ button on ‘Course Detail’ Page.        3. Actor inputs updated information and clicks ‘Save’ button. | 2. System shows a course detail with the following controls:  - ‘Course Name’ text field.  - ‘Description’ text field.  - ‘Start Date’ date selector.  - ‘End Date’ date selector.  - ‘Discount Price’  - ‘Is Extra’  - ‘Is Basic’  - ‘Price’ text field.  - ‘Is Shown’  4. System checks the information.  5. System redirects to Courses Page. |
|
| **Exceptions** | **Actor’s actions:** | **System’s responses:** |
| - Actor does not input Course name.  - Actor does not input Description.  - Actor does not input Price.  - Actor inputs the existing course name. | System displays message to actor:   * ‘The Course Name field is required’. * ‘The Description field is required’. * ‘The Price field is required’. * ‘The course name has existed.’ |

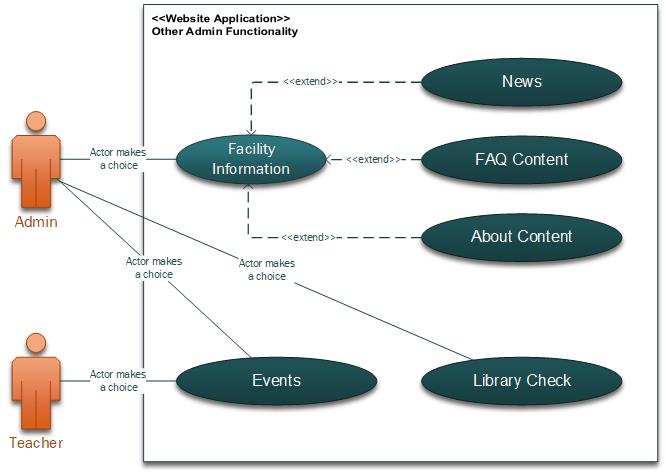
### Registration Management



|  |  |  |
| --- | --- | --- |
| **Author** |  | |
| **Use Case Name** | Take exam | |
| **Actors** | Student | |
| **Description** | Student wants to take exam. | |
| **Requirements** | Student must have an account.  Student must be in a course or register for an examination of an advanced course. | |
| **Pre-conditions** | Actor has logged in. | |
| **Post-conditions** | Success: | |
| Fail: | |
| **Basic flow** | **Actor’s actions:** | **System’s responses:** |
| 1. Actor clicks ‘Take exam’.     3. Actor inputs ‘Password for exam’ and then clicks ‘Go’ button    5.Actor does this exam by choosing a correct answer. When actor have done this exam, clicks ‘Finish’ button | 2. System redirects to Take Exam page and displays form with the following controls:   * ‘Password for exam’ text field. * ‘Go’ button.   4. System shows the exam.  6. System checks all the answers, then shows the result of this exam. |
|
| **Exceptions** | **Actor’s actions:** | **System’s responses:** |
| * Actor left ‘Password for exam’. * Input wrong ‘Password for exam’. * Actor does not click ‘Finish’ button when time out. | System displays message to actor:   * ‘The Password for exam field is required’. * ‘Wrong password for exam’. * System automatically stops what actor is doing, checks answers and shows result. |

## 

## Other Admin Functionalities

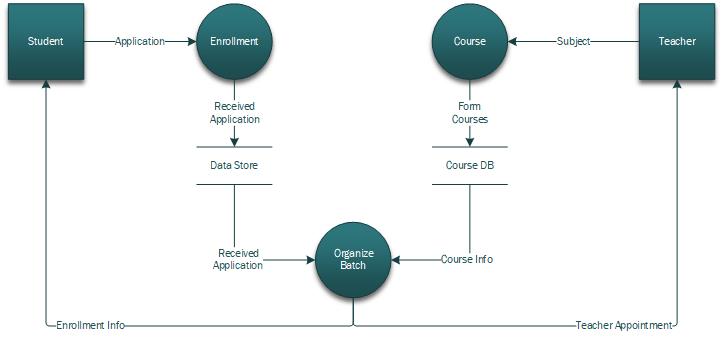


# Data Flow Diagram

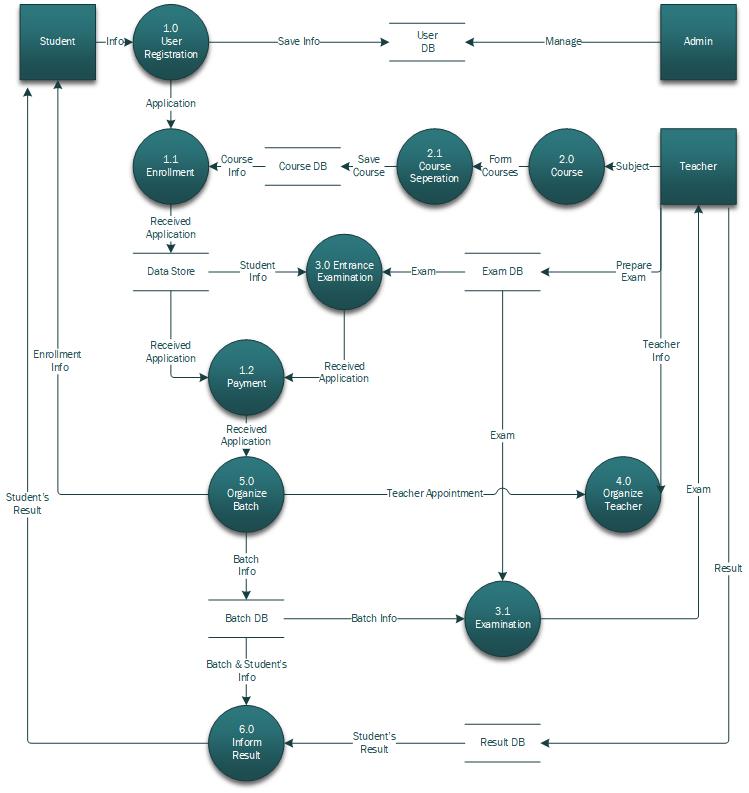
## Context Diagram:



## Level 0 DFD

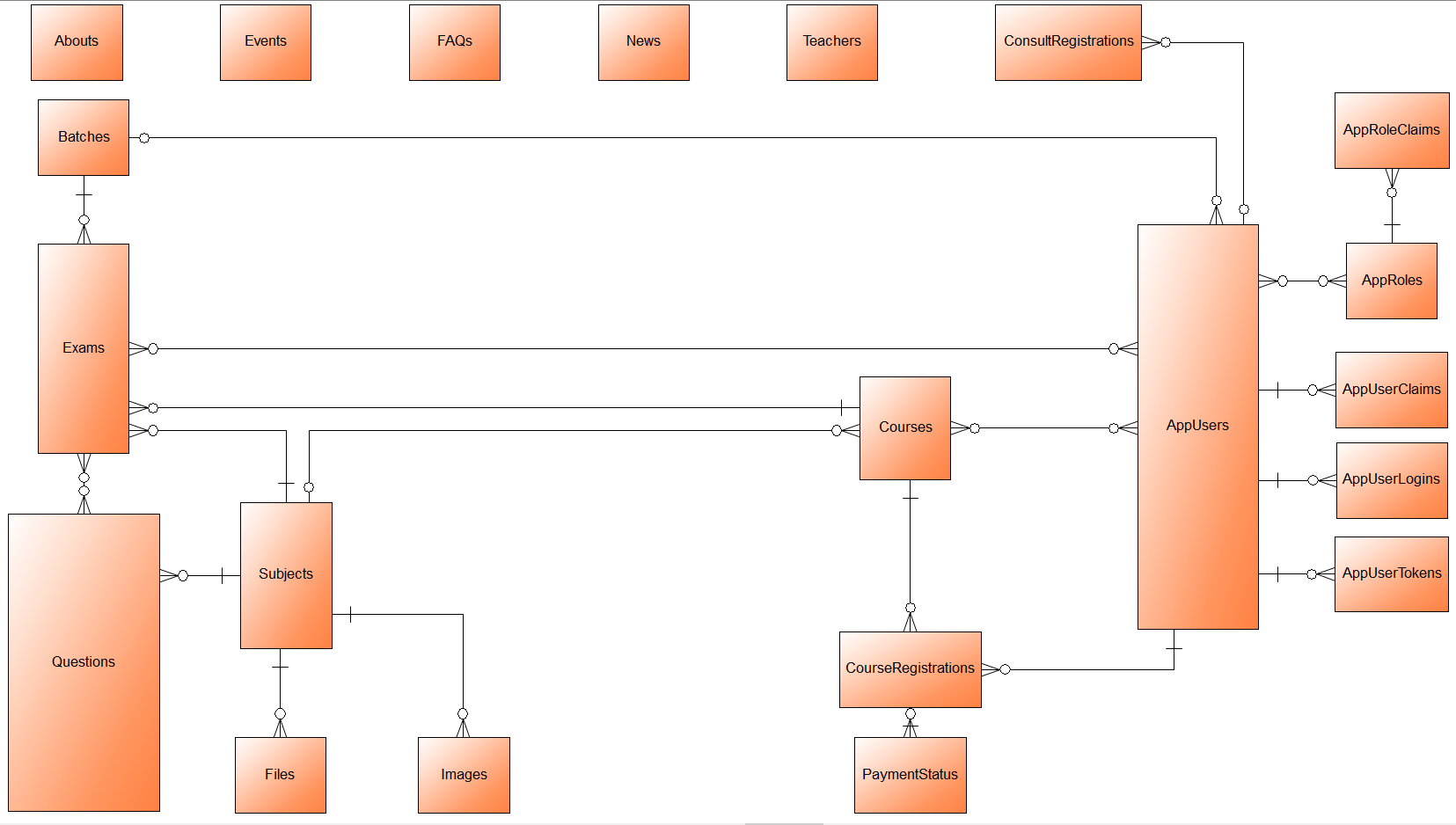


## Level 1 DFD

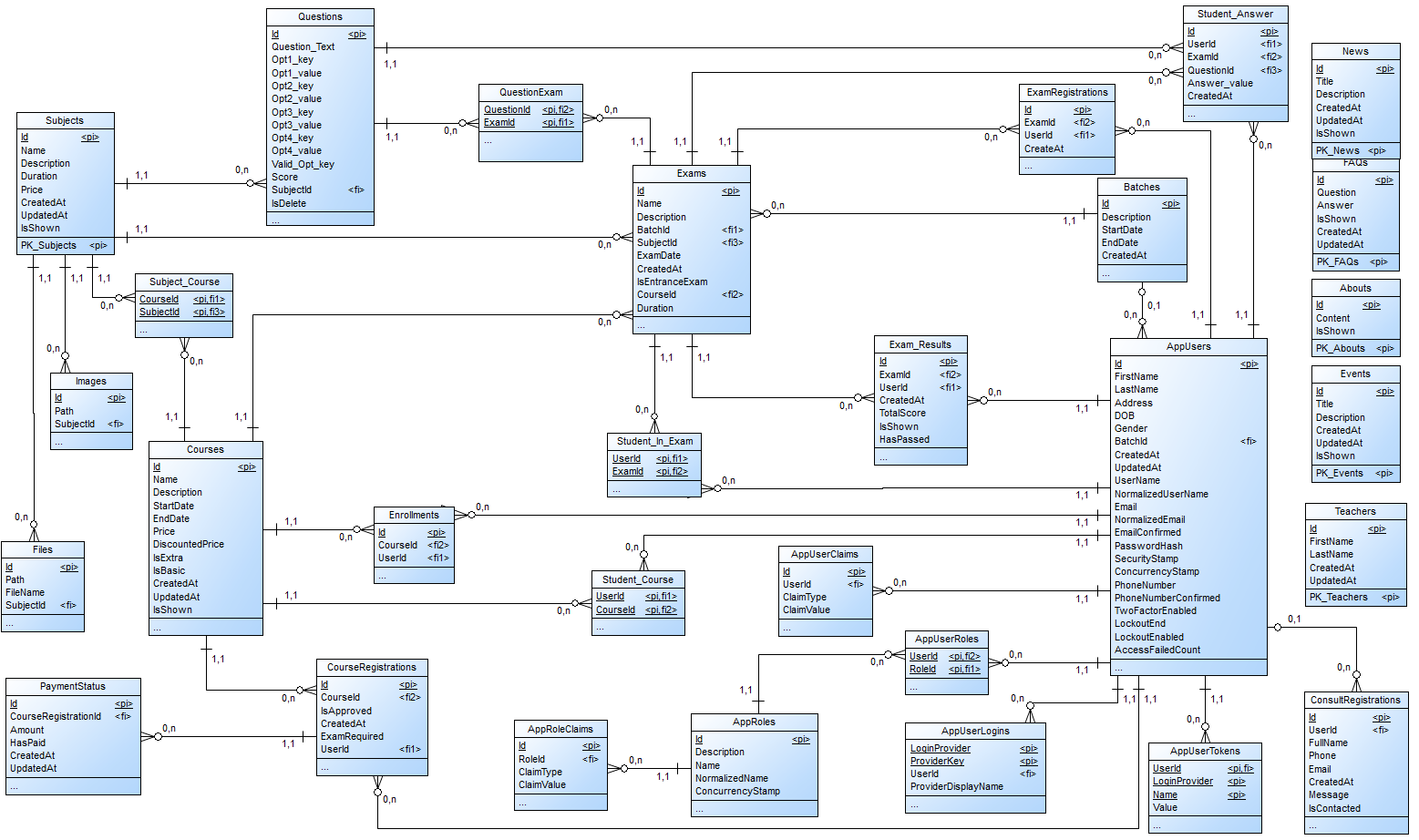
****

# Entity–Relationship Diagram

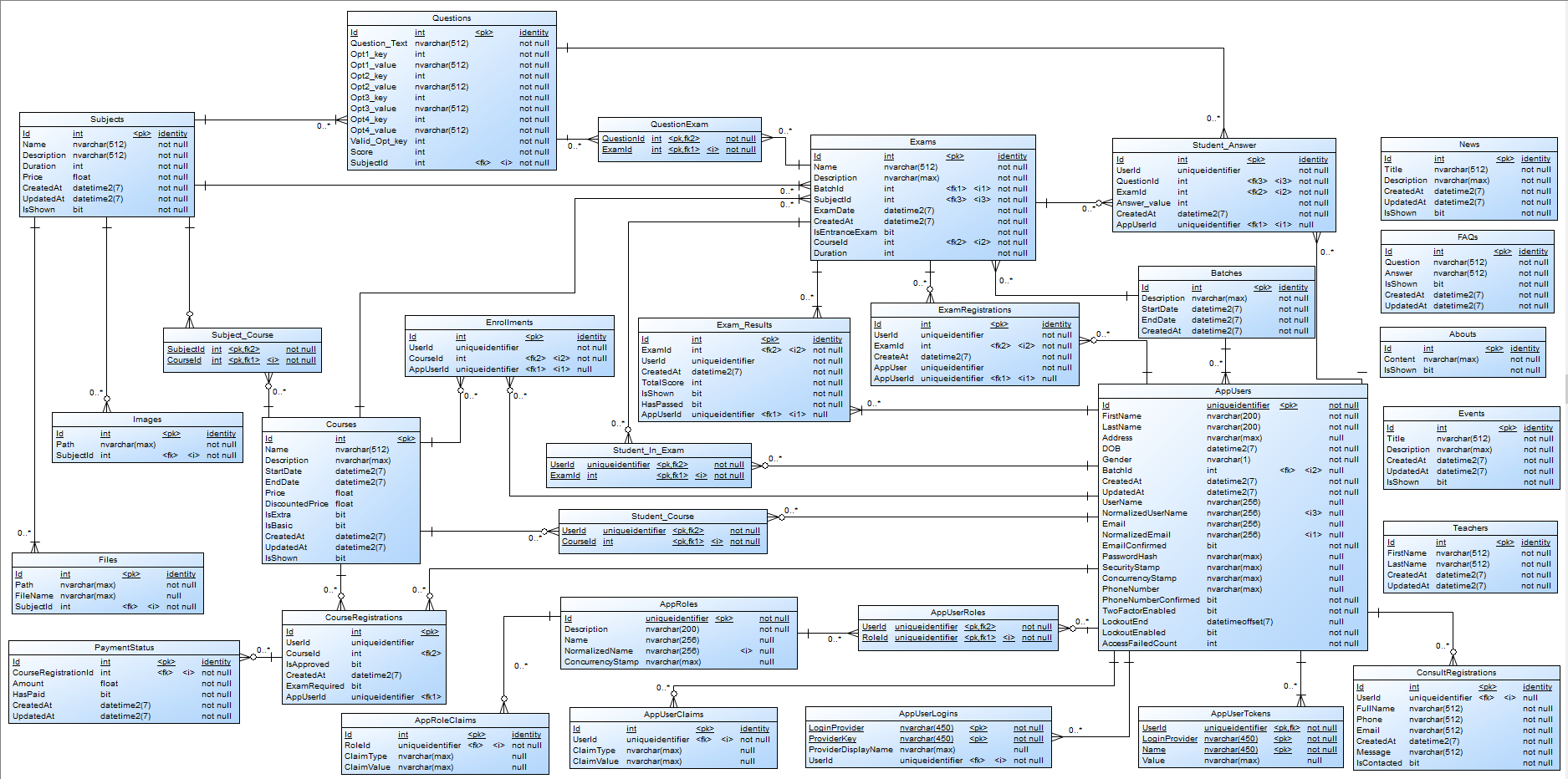
### Conceptual ERD



### Logical ERD

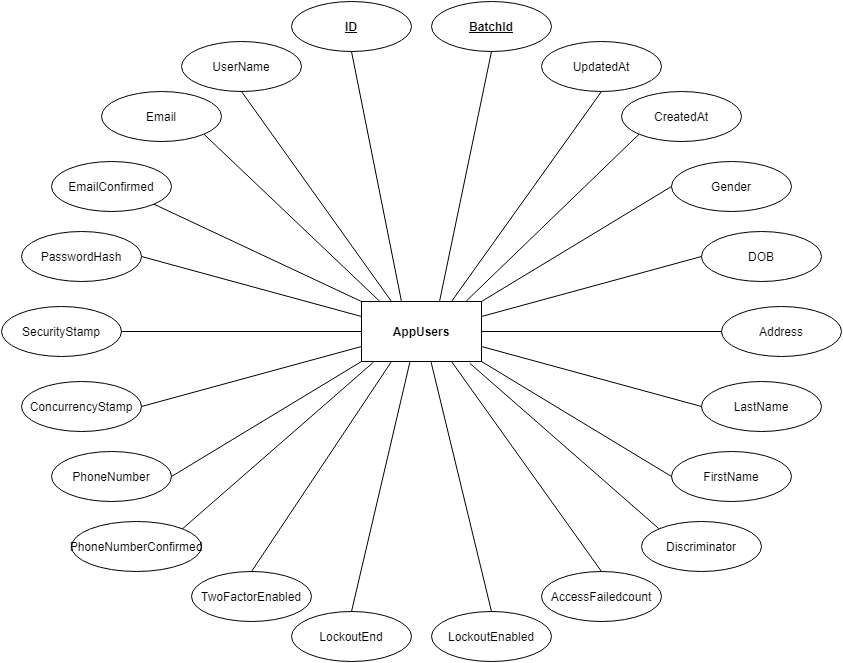


### Physical ERD

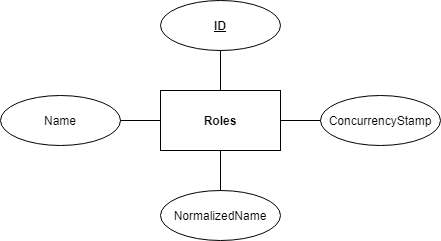


# Entities and Properties

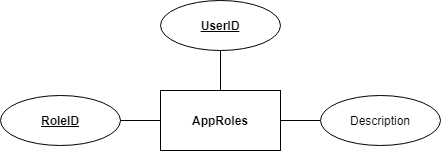
## AppUsers

****

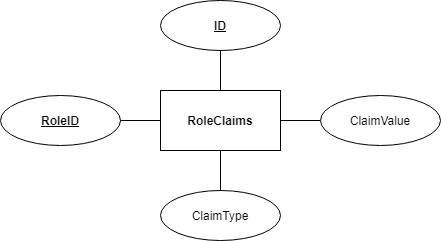
## Roles

****

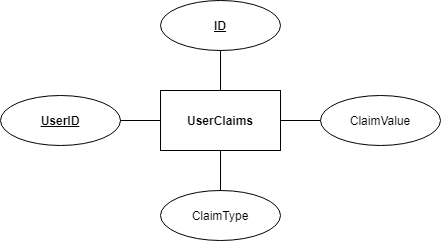
## AppRoles

****

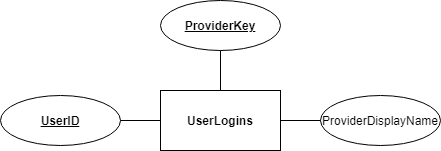
## RoleClaims

****

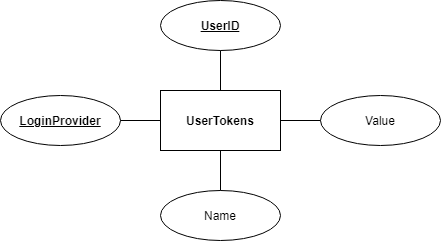
## UserClaims

****

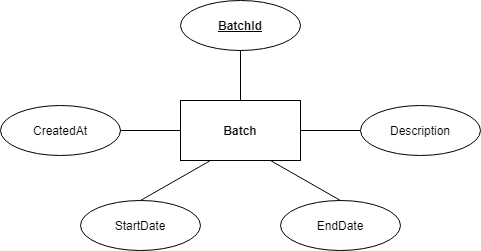
## UserLogins



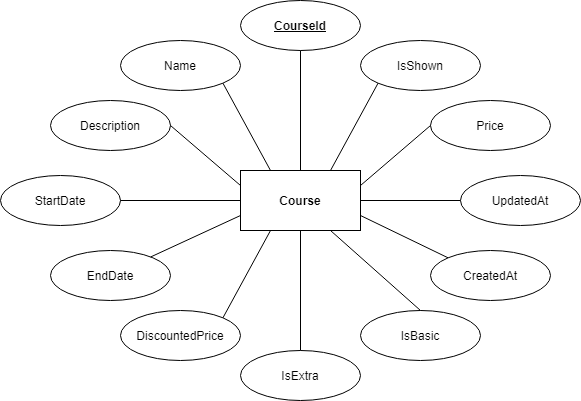
## UserTokens

****

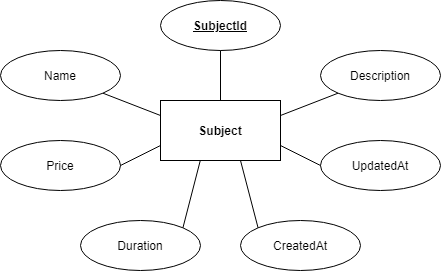
## Batch

****

## Course

****

## Subject

****

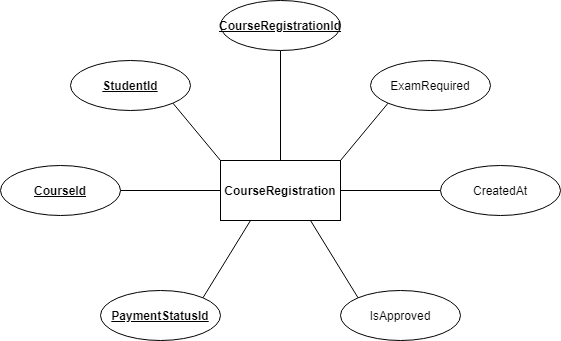
## Subject\_Course

****

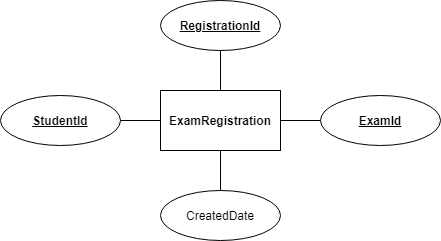
## Exam

## 

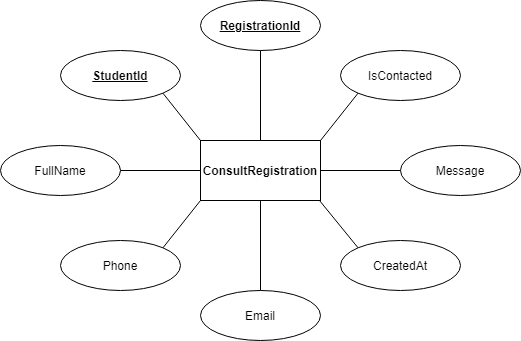
## CourseRegistration

****

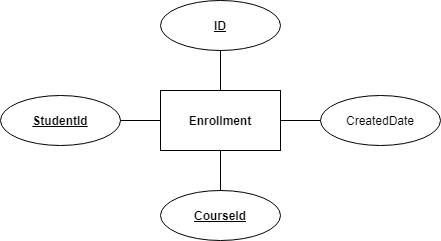
## ExamRegistration



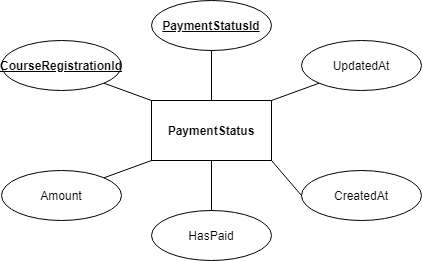
## ConsultRegistration

****

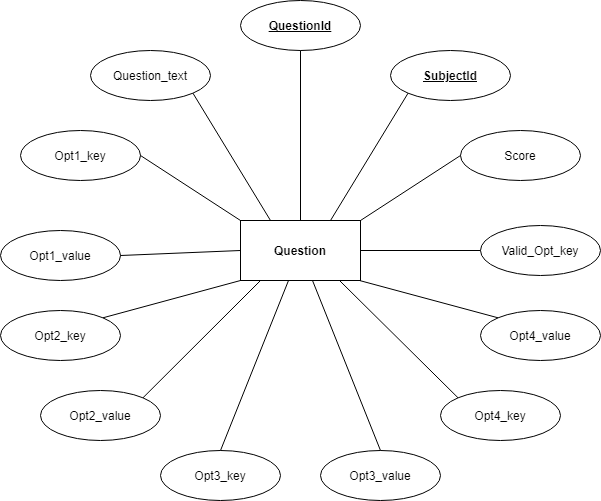
## Enrollment

****

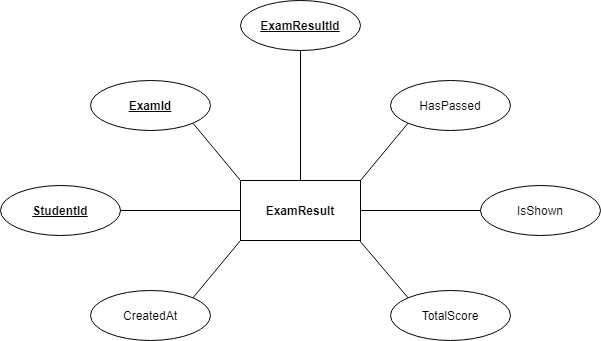
## PaymentStatus

****

1. **Question**

****

1. **ExamResult**

****

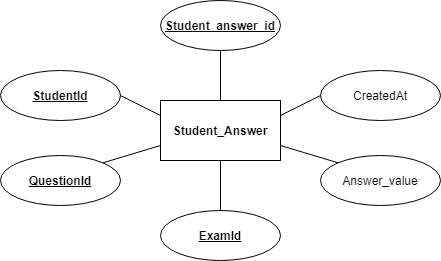
1. **Student\_In\_exam**

****

1. **QuestionExam**

****

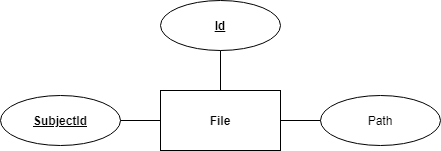
1. **Student\_Answer**

****

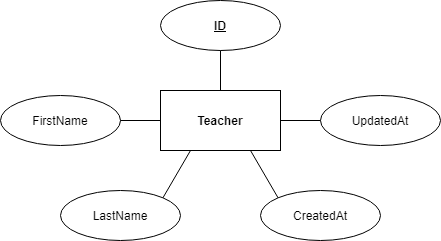
1. **Image**

****

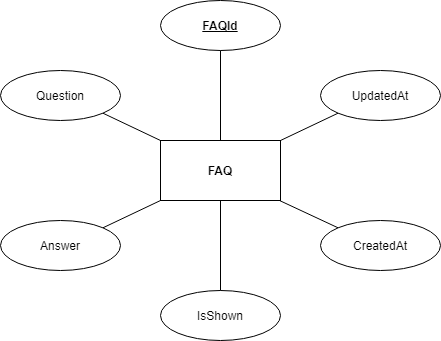
1. **File**

****

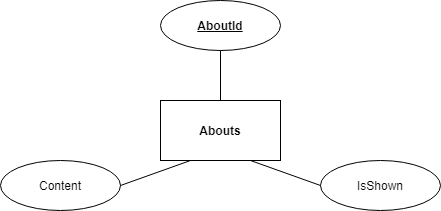
1. **Teacher**

****

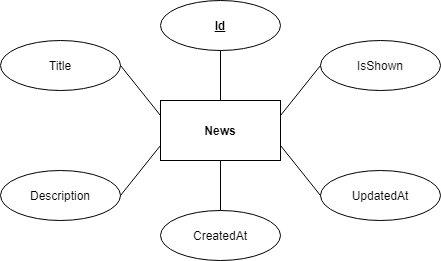
1. **FAQ**

****

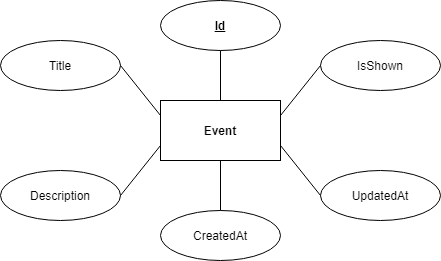
1. **Abouts**

****

1. **News**

****

1. **Event**

****

1. **Table Design**
2. **AppUsers**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table: AppUsers** | | | | |
| **No.** | **Column Name** | **Data Type** | **Constraint** | **Description** |
| 1 | ID | NVARCHAR | NOT NULL PRIMARY KEY | Stores the ID of user |
| 2 | UserName | NVARCHAR | NOT NULL | Stores the name of user |
| 3 | Email | NVARCHAR | NOT NULL | Stores the email of user |
| 4 | EmailConfirmed | BIT | NOT NULL | Confirm the email of user |
| 5 | PasswordHash | NVARCHAR | NOT NULL | Stores the password of user |
| 6 | SecurityStamp | NVARCHAR | NOT NULL |  |
| 7 | ConcurrencyStamp | NVARCHAR | NOT NULL |  |
| 8 | PhoneNumber | NVARCHAR | NOT NULL | Store the phone number of user |
| 9 | PhoneNumberConfirmed | BIT | NOT NULL | Confirm the phone number of user |
| 10 | TwoFactorEnabled | BIT | NOT NULL |  |
| 11 | LockoutEnd | DATETIMEOFFSET | NOT NULL |  |
| 12 | LockoutEnabled | BIT | NOT NULL |  |
| 13 | AccessFailedcount | INT | NOT NULL |  |
| 14 | Discriminator | NVARCHAR | NOT NULL |  |
| 15 | FirstName | NVARCHAR | NOT NULL | Stores the first name of user |
| 16 | LastName | NVARCHAR | NOT NULL | Stores the last name of user |
| 17 | Address | NVARCHAR | NOT NULL | Stores the address of user |
| 18 | DOB | DATETIME | NOT NULL | Stores the date of birth of user |
| 19 | Gender | CHAR | NOT NULL | Stores the gender of user |
| 20 | CreatedAt | DATETIME | NOT NULL |  |
| 21 | UpdatedAt | DATETIME | NOT NULL |  |
| 22 | BatchId | INT |  |  |

1. **Roles**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table: Roles** | | | | |
| **No.** | **Column Name** | **Data Type** | **Constraint** | **Description** |
| 1 | ID | NVARCHAR | NOT NULL PRIMARY KEY | Stores the ID of role |
| 2 | Name | NVARCHAR | NOT NULL | Stores the name of role |
| 3 | NormalizedName | NVARCHAR | NOT NULL | Stores the normalized name of role |
| 4 | ConcurrencyStamp | NVARCHAR | NOT NULL |  |

1. **AppRoles**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table: UserLogins** | | | | |
| **No.** | **Column Name** | **Data Type** | **Constraint** | **Description** |
| 1 | UserID | NVARCHAR | NOT NULL PRIMARY KEY  FOREIGN KEY | Stores the userid of approles |
| 2 | RoleID | NVARCHAR | NOT NULL  PRIMARY KEY  FOEREIGN KEY | Stores the roleid of approles |
| 3 | Description | NVARCHAR | NOT NULL  FOREIGN KEY | Stores the description of approles |

1. **RoleClaims**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table: RoleClaims** | | | | |
| **No.** | **Column Name** | **Data Type** | **Constraint** | **Description** |
| 1 | ID | INT | NOT NULL PRIMARY KEY | Stores the ID of roleclaims |
| 2 | RoleID | NVARCHAR | NOT NULL FOREIGN KEY |  |
| 3 | ClaimType | NVARCHAR | NOT NULL | Stores the claim type of roleclaims |
| 4 | ClaimValue | NVARCHAR | NOT NULL | Stores the claim value of roleclaims |

1. **UserClaims**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table: UserClaims** | | | | |
| **No.** | **Column Name** | **Data Type** | **Constraint** | **Description** |
| 1 | ID | INT | NOT NULL PRIMARY KEY | Stores the ID of userclaims |
| 2 | UserID | NVARCHAR | NOT NULL FOREIGN KEY |  |
| 3 | ClaimType | NVARCHAR | NOT NULL | Stores the claim type of userclaims |
| 4 | ClaimValue | NVARCHAR | NOT NULL | Stores the claim value of userclaims |

1. **UserLogins**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table: UserLogins** | | | | |
| **No.** | **Column Name** | **Data Type** | **Constraint** | **Description** |
| 1 | ProviderKey | NVARCHAR | NOT NULL PRIMARY KEY | Stores the provider key of userlogins |
| 2 | ProviderDisplayName | NVARCHAR | NOT NULL | Stores the providerdisplayname of userlogins |
| 3 | UserID | NVARCHAR | NOT NULL  FOREIGN KEY |  |

1. **UserTokens**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table: UserTokens** | | | | |
| **No.** | **Column Name** | **Data Type** | **Constraint** | **Description** |
| 1 | UserID | NVARCHAR | NOT NULL PRIMARY KEY  FOREIGN KEY | Stores the userid of usertokens |
| 2 | LoginProvider | NVARCHAR | NOT NULL PRIMARY KEY | Stores the loginprovider of usertokens |
| 3 | Name | NVARCHAR | NOT NULL | Stores the name of usertokens |
| 4 | Value | NVARCHAR | NOT NULL | Stores the value of usertokens |

1. **Batch**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table: Batch** | | | | |
| **No.** | **Column Name** | **Data Type** | **Constraint** | **Description** |
| 1 | BatchId | VARCHAR(128) | NOT NULL PRIMARY KEY | Stores the ID of batch |
| 2 | CreatedAt | DATETIME | NOT NULL |  |
| 3 | StartDate | DATE | NOT NULL | Stores the start date of batch |
| 4 | EndDate | DATE | NOT NULL | Stores the end date of batch |
| 5 | Description | VARCHAR(256) |  | Stores the description of batch |

1. **Course**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table: Course** | | | | |
| **No.** | **Column Name** | **Data Type** | **Constraint** | **Description** |
| 1 | CourseId |  | NOT NULL PRIMARY KEY | Stores the ID of course |
| 2 | Name |  | NOT NULL | Stores the name of course |
| 3 | Description |  | NOT NULL | Stores the description of course |
| 4 | StartDate | DATETIME | NOT NULL | Stores the start date of course |
| 5 | EndDate | DATETIME | NOT NULL | Stores the end date of course |
| 6 | DiscountedPrice |  | NOT NULL | Stores the discounted price of course |
| 7 | IsExtra | BOOL |  |  |
| 8 | IsBasic | BOOL |  |  |
| 9 | CreatedAt | DATETIME | NOT NULL |  |
| 10 | UpdatedAt | DATETIME | NOT NULL |  |
| 11 | Price | DOUBLE | NOT NULL | Stores the price of course |
| 12 | IsShown | BOOL |  |  |

1. **Subject**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table: Subject** | | | | |
| **No.** | **Column Name** | **Data Type** | **Constraint** | **Description** |
| 1 | SubjectId |  | NOT NULL PRIMARY KEY | Stores the ID of subject |
| 2 | Name |  | NOT NULL | Stores the name of subject |
| 3 | Price | DOUBLE | NOT NULL | Stores the price of subject |
| 4 | Duration | DOUBLE | NOT NULL | Stores the duration of subject |
| 5 | CreatedAt | DATETIME | NOT NULL |  |
| 6 | UpdatedAt | DATETIME | NOT NULL |  |
| 7 | Description |  | NOT NULL | Stores the description of subject |

1. **Subject\_Course**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table: Abouts** | | | | |
| **No.** | **Column Name** | **Data Type** | **Constraint** | **Description** |
| 1 | SubjectId |  |  |  |
| 2 | CourseId |  |  |  |

1. **Exam**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table: Exam** | | | | |
| **No.** | **Column Name** | **Data Type** | **Constraint** | **Description** |
| 1 | ExamId | INT | NOT NULL PRIMARY KEY | Stores the ID of exam |
| 2 | Name | VARCHAR(256) | NOT NULL | Stores the name of exam |
| 3 | Date | DATE | NOT NULL | Stores the date of exam |
| 4 | Description | VARCHAR(256) | NOT NULL | Stores the description of exam |
| 5 | BatchId | INT | NOT NULL  FOREIGN KEY |  |
| 6 | SubjectId | INT | NOT NULL  FOREIGN KEY |  |
| 7 | CreatedAt | DATETIME | NOT NULL |  |
| 8 | IsEntranceExam | BOOL |  |  |
| 9 | CourseId | INT | NOT NULL  FOREIGN KEY |  |
| 10 | Duration | DOUBLE | NOT NULL | Stores the duration of exam |

1. **CourseRegistration**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table: CourseRegistration** | | | | |
| **No.** | **Column Name** | **Data Type** | **Constraint** | **Description** |
| 1 | CourseRegistrationId |  | NOT NULL PRIMARY KEY | Stores the ID of course registation |
| 2 | StudentId |  | NOT NULL  FOREIGN KEY |  |
| 3 | CourseId |  | NOT NULL  FOREIGN KEY |  |
| 4 | PaymentStatusId |  | NOT NULL  FOREIGN KEY |  |
| 5 | IsApproved | BOOL | NOT NULL |  |
| 6 | CreatedAt | DATETIME | NOT NULL |  |
| 7 | ExamRequired | BOOL | NOT NULL |  |

1. **ExamRegistration**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table: ExamRegistration** | | | | |
| **No.** | **Column Name** | **Data Type** | **Constraint** | **Description** |
| 1 | RegistrationId |  | NOT NULL PRIMARY KEY |  |
| 2 | StudentId |  | NOT NULL  FOREIGN KEY |  |
| 3 | CreatedAt | DATETIME | NOT NULL |  |
| 4 | ExamId |  | NOT NULL  FOREIGN KEY |  |

1. **ConsultRegistration**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table: ConsultRegistration** | | | | |
| **No.** | **Column Name** | **Data Type** | **Constraint** | **Description** |
| 1 | RegistrationId |  | NOT NULL PRIMARY KEY |  |
| 2 | StudentId | NULLABLE | FOREIGN KEY |  |
| 3 | FullName |  | NOT NULL |  |
| 4 | Phone |  | NOT NULL |  |
| 5 | Email |  | NOT NULL |  |
| 6 | CreatedAt |  | NOT NULL |  |
| 7 | Message | NULLABLE |  | Stores the message of consult registration |
| 8 | IsContacted | BOOL |  |  |

1. **Enrollment**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table: Enrollment** | | | | |
| **No.** | **Column Name** | **Data Type** | **Constraint** | **Description** |
| 1 | Id | INT | NOT NULL PRIMARY KEY | Stores the ID of enrollment |
| 2 | StudentId | INT | NOT NULL  FOREIGN KEY |  |
| 3 | CourseId | INT | NOT NULL  FOREIGN KEY |  |
| 4 | CreatedDate | DATE | NOT NULL | Stores the created date of enrollment |

1. **PaymentStatus**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table: PaymentStatus** | | | | |
| **No.** | **Column Name** | **Data Type** | **Constraint** | **Description** |
| 1 | PaymentStatusId | INT | NOT NULL PRIMARY KEY | Stores the ID of payment status |
| 2 | CourseRegistrationId | INT | NOT NULL  FOREIGN KEY | Stores the ID of course |
| 3 | Amount | FLOAT | NOT NULL | Stores the amount of payment |
| 4 | HasPaid | BOOL | NOT NULL | Stores the ID of exam |
| 5 | CreatedAt |  | NOT NULL |  |
| 6 | UpdatedAt |  | NOT NULL |  |

1. **Question**

|  |  |  |  |
| --- | --- | --- | --- |
| **Table: Question** | | | |
| **Column Name** | **Data Type** | **Constraint** | **Description** |
| QuestionId | INT | NOT NULL PRIMARY KEY | Stores the ID of question |
| Question\_text | VARCHAR(512) | NOT NULL | Stores the text of question |
| Opt1\_key | INT | NOT NULL | Stores the option1 key of question |
| Opt1\_\_value | VARCHAR(512) | NOT NULL | Stores the option1 value of question |
| Opt2\_key | INT | NOT NULL | Stores the option2 key of question |
| Opt2\_value | VARCHAR(512) | NOT NULL | Stores the option2 value of question |
| Opt3\_key | INT | NOT NULL | Stores the option3 key of question |
| Opt3\_value | VARCHAR(512) |  | Stores the option3 value of question |
| Opt4\_key | INT | NOT NULL | Stores the option4 key of question |
| Opt4\_value | VARCHAR(512) | NOT NULL | Stores the option4 value of question |
| Valid\_Opt\_key | INT | NOT NULL | Stores the valid option key of question |
| Score | INT | NOT NULL | Stores the score of question |
| SubjectId | INT | NOT NULL | Stores the id of subject |

1. **Exam\_result**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table: Exam\_result** | | | | |
| **No.** | **Column Name** | **Data Type** | **Constraint** | **Description** |
| 1 | ExamResultId | INT | NOT NULL PRIMARY KEY | Stores the ID of exam result |
| 2 | ExamId | INT | NOT NULL  FOREIGN KEY |  |
| 3 | StudentId | INT | NOT NULL  FOREIGN KEY |  |
| 4 | CreatedAt | INT | NOT NULL |  |
| 5 | TotalScore | INT | NOT NULL | Stores the total score of exam result |
| 6 | IsShown | BOOL | NOT NULL |  |
| 7 | HasPassed | BOOL | NOT NULL |  |

1. **Student\_In\_exam**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table: Student\_In\_exam** | | | | |
| **No.** | **Column Name** | **Data Type** | **Constraint** | **Description** |
| 1 | ExamId | INT | NOT NULL PRIMARY KEY |  |
| 2 | UserId |  | NOT NULL |  |

1. **QuestionExam**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table: QuestionExam** | | | | |
| **No.** | **Column Name** | **Data Type** | **Constraint** | **Description** |
| 1 | QuestionId | INT | NOT NULL PRIMARY KEY |  |
| 2 | ExamId | INT | NOT NULL |  |

1. **Student\_Answer**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table: Student\_Answer** | | | | |
| **No.** | **Column Name** | **Data Type** | **Constraint** | **Description** |
| 1 | Student\_answer\_id | INT | NOT NULL PRIMARY KEY |  |
| 2 | StudentId |  | NOT NULL  FOREIGN KEY |  |
| 3 | Question\_id | INT | NOT NULL |  |
| 4 | Exam\_id | INT | NOT NULL |  |
| 5 | Answer\_value | INT |  |  |
| 6 | CreatedAt |  |  |  |

1. **Image**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table: Image** | | | | |
| **No.** | **Column Name** | **Data Type** | **Constraint** | **Description** |
| 1 | ImageId |  | NOT NULL PRIMARY KEY | Stores the ID of image |
| 2 | Path |  | NOT NULL | Stores the path of image |

1. **File**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table: File** | | | | |
| **No.** | **Column Name** | **Data Type** | **Constraint** | **Description** |
| 1 | ID |  | NOT NULL PRIMARY KEY | Stores the ID of file |
| 2 | Path |  | NOT NULL | Stores the path of file |
| 3 | SubjectId |  | NOT NULL  FOREIGN KEY |  |

1. **Teacher**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table: Teacher** | | | | |
| **No.** | **Column Name** | **Data Type** | **Constraint** | **Description** |
| 1 | ID |  | NOT NULL PRIMARY KEY | Stores the ID of teacher |
| 2 | FirstName |  | NOT NULL | Stores the first name of teacher |
| 3 | LastName |  | NOT NULL | Stores the last name of teacher |
| 4 | CreatedAt | DATETIME | NOT NULL |  |
| 5 | UpdatedAt | DATETIME | NOT NULL |  |

1. **FAQ**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table: FAQ** | | | | |
| **No.** | **Column Name** | **Data Type** | **Constraint** | **Description** |
| 1 | FAQId |  | NOT NULL PRIMARY KEY | Stores the ID of FAQ |
| 2 | Question | NVARCHAR | NOT NULL | Stores the question of FAQ |
| 3 | Answer | NVARCHAR | NOT NULL | Stores the answer of FAQ |
| 4 | IsShown | BOOL | NOT NULL |  |
| 5 | CreatedAt | DATETIME | NOT NULL |  |
| 6 | UpdatedAt | DATETIME | NOT NULL |  |

1. **Abouts**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table: Abouts** | | | | |
| **No.** | **Column Name** | **Data Type** | **Constraint** | **Description** |
| 1 | AboutId |  | NOT NULL PRIMARY KEY | Stores the ID of about |
| 2 | Content | NVARCHAR | NOT NULL | Stores the content of about |
| 3 | IsShown |  |  |  |

1. **News**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table: Abouts** | | | | |
| **No.** | **Column Name** | **Data Type** | **Constraint** | **Description** |
| 1 | Id |  | NOT NULL PRIMARY KEY | Stores the ID of news |
| 2 | Title |  | NOT NULL | Stores the title of news |
| 3 | Description |  |  | Stores the description of news |
| 4 | CreatedAt | DATETIME |  |  |
| 5 | UpdatedAt | DATETIME |  |  |
| 6 | IsShown |  |  |  |

1. **Event**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table: Abouts** | | | | |
| **No.** | **Column Name** | **Data Type** | **Constraint** | **Description** |
| 1 | Id |  | NOT NULL PRIMARY KEY | Stores the ID of event |
| 2 | Title |  | NOT NULL | Stores the title of event |
| 3 | Description |  |  | Stores the description of event |
| 4 | CreatedAt | DATETIME |  |  |
| 5 | UpdatedAt | DATETIME |  |  |
| 6 | IsShown |  |  |  |

**Task sheet review 2**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Project Ref. No: 4 | Project Title:  Railway Reservation Manage System | Date of Preparation of Activity Plan | | | |
| No. | Task | Actual Start Date | Actual Days | Team Member Name | Status |
| 01 | Architecture & Design of the Project | Nov 8, 2019 | 6 | All Members | Completed |
| 02 | Algorithms - Data Flowchart |  | Completed |
| 03 | Data Flow Diagram |  | Completed |
| 04 | Use Case Diagram |  | Completed |
| 05 | ERD |  | Completed |

|  |  |  |
| --- | --- | --- |
|  | Prepare By: Group 1 | Approved By: Faculty |
| Date: Oct 02, 2021 | Team Leader  Nguyen Thanh Trung | Mr. Ngo Phuoc Nguyen |